CPSC 359 – Fall 2019
Assignment 4: RPi Frame Buffer and SNES Controller

based on L. Manzara 2019

due: 1700h 06-Dec-2019

Background

This assignment is a variation of the one created by Dr. Leonard Manzara for CPSC 359, Winter 2019.

In this assignment you will use the Raspberry Pi (RPi) frame buffer to create a visual display. You will also use the GPIO capabilities of the Pi to operate an external device, a Super Nintendo Entertainment System (SNES) controller (`https://en.wikipedia.org/wiki/Super_Nintendo_Entertainment_System`).

Objective

Your goal is to write a program, in *C*, that emulates an *Etch A Sketch*, a retro toy from the 1960s (`https://en.wikipedia.org/wiki/Etch_A_Sketch`). *Etch A Sketch* allowed people to make rudimentary drawings with a small set of simple functions. Although *Etch A Sketch* was a mechanical device, you can emulate its functions with software on the RPi.

Etch A Sketch had three controls summarized in Table 1. Note that the *Etch A Sketch* was like a plot that could not lift its pen. So from a default starting position, the user moves the "pen" up/down/left/right to draw a picture. The user can restart at any time by inverting the *Etch A Sketch* and shaking it to erase. After erasing drawing continues from the last position of the pen.

Etch A Sketch Control	Description	SNES Implementation
Horizontal knob	Move drawing position left and right	Left and right joystick buttons
Vertical knob	Move drawing position up and down	Up and down joystick buttons
Invert/shake	Erase image	Start button

Table 1: Summary of *Etch A Sketch* controls.

Details/Hints

You may use code from examples provided in the course material through D2L – but be sure to note in our source code comments where the code came from.

You should use the system timer counter register to time your delays. See the Broadcom ARM Peripherals manual, p 172. It is not more efficient than than using delay loops, but it does expose you to another, easy-to-use capability of the Raspberry Pi. You are not required to use interrupts for this assignment.

You may notice that writing pixels to the frame buffer is slow – it takes almost one second to fill the screen. Don't worry about it. It is beyond the scope of this course to fix the problem.

Although not strictly necessary for this assignment, I found it helpful to implement a line drawing function. You may want to look at *Bresenham's line algorithm*. If you use an implementation that you find online, be sure to cite the source.

As stated above, use the *C* language because: it will give you more practice with an important programming language, and it is much easier to work with then assembly language. In fact, this kind of low-level machine programming is precisely what *C* does best. See `https://en.wikipedia.org/wiki/C_(programming_language)`.

As with any programming project, I recommend that you approach this incrementally. Start by implementing a simple subset of the required functionality, then add the remaining functionality one piece at a time. I suggest doing the interrupts last.

Debugging is hard to do in this bare metal environment. Doing incremental changes will help you to zero on the source of errors. Also, UART output may help to find some bugs, so set that up.

Good coding practices include all the good habits you should exhibit when writing your code. These include, but are not limited to:

- indentation,
- informative variable and function names,
- helpful comments,
- the presence additional, helpful files such as readme.txt and Makefile, and
- code modularity.

Deliverables

Turn in the following items for evaluation.

1. All files required to build your program. This includes all of your source code files, Makefile, and a readme.txt file with instructions.
2. A demo of your working program for your TA during a tutorial session – date to be determined by your TA.

Bonus (10%)

For a 10% bonus, you can implement a flood fill algorithm. I used the SNES *X* button to initiate the fill. This should be a fairly obvious algorithm if you use what you learned in CPSC 331, but if you look anything up, be sure to cite you sources.

Evaluation

1	Program runs as specified	
	Drawing controls work	5pts
	Erase works	5pts
2	Configuration of I/O pins	3pts
3	Configuration of frame buffer	4pts
4	Coding practices	3pts
5	Flood fill (bonus)	2pts
	Total	20pts

There is no team work. Each students must submit individual solutions.

Late work

After the deadline and up to 24hrs late: -5pts. After 24hrs and up to 48hrs late: -10pts. Over 48hrs late: -20pts, i.e., no assignment will be accepted beyond 48hrs after the deadline.

Plagiarism

Work must me the sole work of the individual submitting the work.

You may find that the this assignment is similar to other published code. Nevertheless, you should go through the process yourself, and submit your own work. If you do borrow from other sources, cite the source and clearly indicate what you have borrowed, keeping in mind the design must be substantially your own. If you cite your sources, worst case you may receive a reduced grade for borrowing too much. If you borrow, but do not cite, that is plagiarism and academic misconduct, and carries severe penalties as determined by the Faculty of Science. You may use code provided in the course content on D2L, but you must still cite the source.

As a guideline, consider the 20-minute rule. Talk with your colleagues and consult other sources (cite them please). Wait at least 20 minutes, then do your work to be sure that it is your own. Less then 20 minutes usually means that you are merely copying work from the original source.